FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Review report of a final thesis

Reviewer:	Ing. Josef Kokeš
Student:	Erich Winkler
Thesis title:	Java Card Second-factor Authentication Plugin for KeePass
Branch / specialization:	Computer Security and Information technology
Created on:	January 19, 2022

Evaluation criteria

1. Fulfillment of the assignment

[1] assignment fulfilled
[2] assignment fulfilled with minor objections
▸ **[3] assignment fulfilled with major objections**
[4] assignment not fulfilled

Technically, all instructions of the assignment were fulfilled, but with a different focus than that of the stated overall goal. The thesis works as a proof of concept of implementing a second authentication factor into KeePass, but it fails as a solution to the goal of "develop a plugin" (which I understand to mean "plugin that can actually be used in practice"). I am told the former was actually what both the supervisor and the student intended to do, but it's the latter that I understand to be the official assignment.

2. Main written part 50/100 (E)

The thesis is written in English. The language level starts quite high, though not perfect, but tends to decrease as the work progresses. Nevertheless, the thesis remains understandable throughout, even if it quickly starts to annoy the reader with the use of pronoun "you". The formal aspects are fine as well, except that acronyms are used without ever being explained (it's not sufficient to list them in the list of acronyms) and that in the later parts of the thesis, successive section headers appear without any text in between them.

I do notice quite a few issues with the factual accuracy of the content, though, particularly with the security aspects.

First of all, the student's claim that the inherent authentication factors are the most reliable ones should be substantiated with a reference and properly established within the work. It is a quite common complaint, for example, that these factors are often easy to copy and difficult to change, which makes them problematic when used in the context of a password manager.

I feel that the threat analysis needs a significant overhaul, too. When I first saw the threat model, I thought that the student was on the right track, but when explored in more detail, the model turns out to be rather incomplete and for the most part it is not acted upon by the rest of the thesis anyway. As far as I can tell, the student assumed that encrypting the data during transfer would solve all of the threats, but that is definitely not the case (e.g. by modifying the plugin the attacker could defeat this protection and create a fake card; the attacker would not even need to do that, it they managed to capture the decrypted output of the card once, it would be sufficient to decrypt the password database any time).

I don't understand the student's use of RSA encryption. The student is aware of the man-in-the-middle attack (page 27), but still implements a "protection" that consists of an unauthenticated sending of the RSA public key to the card so that the card can encrypt the response data with it? Not to mention other aspects of a secure use of RSA such as using proper padding. I would not be surprised if this use of encryption literally reduced security!

The possibility (or impossibility) or reading the card's memory is not mentioned at all and definitely should, considering that PIN is stored as plaintext (not even a plain hash!) - e.g. Hogenboom, Mostowski: Full Memory Read Attack on a Java Card. Incidentally, I find it strange that not a single article on the security of smartcards in general or javacards in particular is referenced from the thesis.

Many more possible attack vectors are left out from the thesis. Practical concerns such as how to protect against data loss in case of a lost or damaged card or a damaged user account in Windows are not discussed either.

Following up on the comments from the previous section of this evaluation, I am given to understand that the focus of the thesis was actually different from what I understood from the assignment. That would explain why the student spends too little time on sections which I consider important (such as 3.3 or 5) and quite a lot on sections that I feel could be significantly reduced (2.3, 3.2, 3.4) - if the goal was to create a proof-of-concept of adding a second authentication factor to KeePass no matter what real security benefit (if any) is gained through that, the text would be adequate. If the goal was to create a useful plugin, the text needs a deep overhaul.

3. Non-written part, attachments 25/100 (F)

It is very difficult to judge the code due to the confusion about what is the actual goal of the thesis. If the goal was to create a proof-of-concept, the submitted code is adequate. If the goal was to create a usable plugin, the code fails, especially in its security aspects - disregard for the moment whether its design is correct, but it's not implemented correctly either: The code rarely checks input values, even if it has strict requirements on them - e.g. the symmetric shared keys must be of a particular size, otherwise they can be encrypted but not decrypted; it is assumed that the RSA-encrypted data will always be of a particular size which is not guaranteed even if we use the expected size of the key; missing files will cause the application to crash; secure deletion of sensitive data is not performed, etc.

Regardless of the actual goal of the thesis, the code is also extremely disorganized. The CD contains the text of the thesis without its source code (is this even permitted?) and three directories with the components of the implementation. While the directories are described, their content is not, and it needs to be - we get source code mixed with binaries, test files, config and control files of the development environment (I think), relics of the build process, etc. Filenames are not descriptive at all (the Java applet for the card is called FinalTest for some reason) and frequently there's no clear relationship between them (e.g. in the EncryptSecret directory we get one source code file, three EXE files and two DLLs; the DLLs and one of the EXEs were probably built from some version of the source code, the two remaining EXEs seem to be wrappers to call the DLL from a native code, although I have no idea why that is done since KeePass can certainly import the dotnet DLLs directly). One of the directories contains a Documentation subdirectory which seems to contain dummy data, the others are completely undocumented.

Overall, the code does work if the conditions are right, but it's not production-quality. It very much looks like the student ran out of time while debugging so he burned what he had at that time to the CD and submitted it - but I know that is not true. The only reasonable conclusion can be that the code was never meant to be used in practice and instead served only as a proof of concept. That could be considered acceptable, but I feel that even a proof-of-concept code submitted as a part of thesis should not ignore security completely, especially when the student's branch of study is focused on security.

4. Evaluation of results, publication outputs and awards 50/100 (E)

The thesis as a whole can serve as a proof-of-concept of adding a second authentication factor to a password manager. It demonstrates how several relevant issues of that can be handled and as such can act as an acceptable starting point or perhaps a point of reference. It emphatically does not work as a solution that can be immediately (or even with minor modifications) used - partially because of the code quality issues outlined above, but even more significantly because the security analysis is not reliable and as a result it's quite possible that the code is trying to solve the wrong thing - meaning that even if the code were perfect, it might still not be usable in practice.

The overall evaluation 40/100 (F)

Again, just as the unclear objectives made it difficult to evaluate the component parts of the thesis, they make it difficult to evaluate it as a whole. If the purpose was to create a functional plugin that would allow the user to add a second factor to KeePass, it fails - the security analysis is inadequate and as a result, the code is at best unreliable, more likely actually reduces security (due to several aspects, including a much higher risk of data loss without getting any security benefits in return). If the purpose was to demonstrate how would one go about implementing a second authentication factor in KeePass, it works, provided that the reader understands that they should create their own security analysis and that they must re-write the code from scratch. It is my understanding that the latter was actually the case, but since as a reviewer I must honor the work (including its assignment) as submitted, I have no choice but to not recommend the thesis for a defense and grade it F-Failed. Should the State Exam Committee feel that the goal of the thesis really was to create a proof-of-concept, a grade of D seems reasonable, given the deficiencies in both the textual and non-textual part of the work.

Questions for the defense

1) What was the actual objective of the thesis?
2) What security benefits do you expect from the use of RSA in your code?
3) What are the conditions that must be met for your solution to actually increase the security of KeePass?

Instructions

Fulfillment of the assignment

Assess whether the submitted FT defines the objectives sufficiently and in line with the assignment; whether the objectives are formulated correctly and fulfilled sufficiently. In the comment, specify the points of the assignment that have not been met, assess the severity, impact, and, if appropriate, also the cause of the deficiencies. If the assignment differs substantially from the standards for the FT or if the student has developed the FT beyond the assignment, describe the way it got reflected on the quality of the assignment's fulfilment and the way it affected your final evaluation.

Main written part

Evaluate whether the extent of the FT is adequate to its content and scope: are all the parts of the FT contentful and necessary? Next, consider whether the submitted FT is actually correct – are there factual errors or inaccuracies?

Evaluate the logical structure of the FT, the thematic flow between chapters and whether the text is comprehensible to the reader. Assess whether the formal notations in the FT are used correctly. Assess the typographic and language aspects of the FT, follow the Dean's Directive No. 52/2021, Art. 3.

Evaluate whether the relevant sources are properly used, quoted and cited. Verify that all quotes are properly distinguished from the results achieved in the FT, thus, that the citation ethics has not been violated and that the citations are complete and in accordance with citation practices and standards. Finally, evaluate whether the software and other copyrighted works have been used in accordance with their license terms.

Non-written part, attachments

Depending on the nature of the FT, comment on the non-written part of the thesis. For example: SW work – the overall quality of the program. Is the technology used (from the development to deployment) suitable and adequate? HW – functional sample. Evaluate the technology and tools used. Research and experimental work – repeatability of the experiment.

Evaluation of results, publication outputs and awards

Depending on the nature of the thesis, estimate whether the thesis results could be deployed in practice; alternatively, evaluate whether the results of the FT extend the already published/known results or whether they bring in completely new findings.

The overall evaluation

Summarize which of the aspects of the FT affected your grading process the most. The overall grade does not need to be an arithmetic mean (or other value) calculated from the evaluation in the previous criteria. Generally, a well-fulfilled assignment is assessed by grade A.